Limits of Parallelism and Boosting in Dim Silicon

The authors investigate the limit of voltage scaling together with task parallelization to maintain task-completion latency while reducing energy consumption. They examine improvements in energy efficiency and parallelism due to quickly boosting a core's operating voltage. When accounting for parallelization overheads, minimum task energy is obtained at near-threshold supply voltages across six commercial technology nodes and provides a roughly 4× improvement in overall CMP energy efficiency.

Nathaniel Pinckney
Ronald G. Dreslinski
Korey Sewell
David Fick
Trevor Mudge
Dennis Sylvester
David Blaauw
University of Michigan

Conventional voltage scaling has slowed in recent years, limiting processor frequency to meet power-density constraints. Consequently, processor designs have added more cores without significantly increasing frequency, leading to a prevalence of chip multiprocessors (CMP)[1] in contemporary commercial architectures. However, because the number of cores has been increasing geometrically with each process node while die area has remained fixed, the total chip power has again started to increase, despite relatively flat core frequencies. In practice, the maximum allowable power dissipation of a single die is constrained by thermal-cooling limits. Hence, the consequence of supply-voltage stagnation is a limit on the number of cores that can be active simultaneously on a die and, thus, a limit on the attainable performance of a modern CMP.

The problem of power-constrained core underutilization has been observed recently in the literature and is sometimes referred to as *dark silicon*.[2] As a result, the most recent server-class CMPs incorporate extensive power-gating methods to turn off idle cores to free the thermal budget for active cores.[3] Because modern CMP performance is now limited by power and not die area, a paradigm shift is needed in CMP design: cores are plentiful, but power for them is not.

The most effective knob for reducing the energy consumption of a task running on a microprocessor is lowering the operating voltage.[4] In tandem, processor frequency is reduced and task-completion latency is increased. To address this, parallelization can be used to counteract lower clock frequencies and maintain latency. In this approach, the task's execution code is parallelized so that the task executes on multiple cores in the CMPs, each operating at a lower frequency and voltage, leading to the term *dim silicon*.[5] In this way, the completion time remains the same as when the same task is executed serially on a single core at full voltage, while total energy for completing the task is reduced. This reduction in energy consumption in turn allows more tasks to be executed on the CMP, thereby increasing overall CMP task throughput.

This combined voltage/parallelization approach is similar to the simpler circuit-based parallelization approach proposed previously,[6]

Published by the IEEE Computer Society

0272-1732/13/$31.00 © 2013 IEEE

which trades off energy for latency. The method envisioned here instead parallelizes the algorithm and thereby maintains task latency while still obtaining energy improvement.[7] However, parallelization entails several overheads, which limit the obtainable energy improvement from the proposed approach. Hence, there exists a minimum energy point, at which a task is optimally parallelized and voltage scaling reaches its efficiency limit.

In this article, we study this energy minimum and its associated energy gains, core operating voltage, and task parallelization. Additionally, we study the impact of raising the supply voltage, by boosting cores to overcome serial portions of code, which reduces overhead and allows for further parallelism.

Scaling limiters

Three key factors limit energy-efficient scaling when a task is parallelized to maintain constant latency: leakage, Amdahl overheads, and architectural overheads. Each of these limiters contributes to increased minimum energy and raises the energy-efficient operating point, V_{opt}. In a system with boosting, Amdahl overheads can be eliminated and V_{opt} lowered. We analyze the three key limiters in the following subsections.

Leakage

We will assume that a task can be perfectly parallelized across cores to compensate for frequency loss at a lower voltage and that the only nonideality from running at a slower clock frequency is transistor leakage. Reducing the supply voltage initially increases energy efficiency of a computation quadratically, thereby yielding dramatic energy-efficiency gains.[8] In the past seven years, leakage energy has been shown to pose a fundamental limiting factor to energy-efficiency gains through voltage reduction.[9,10] The required energy to complete a task can be divided into two categories, *dynamic* and *static*. The classic relationship between energy and operating voltage is:

$$\begin{aligned} E_{\mathrm{total}} &= E_{\mathrm{dynamic}} + E_{\mathrm{static}} \\ &= CV_{\mathrm{DD}}^2 + I_{\mathrm{leak}} V_{\mathrm{DD}} T_{\mathrm{task}} \end{aligned}$$

Dynamic or active energy is the energy consumed in charging and discharging the transistor and interconnect capacitances associated with the task being executed. Static or leakage energy results from the always-present subthreshold and gate oxide currents integrated over the time T_{task} to complete a task. Whereas dynamic energy represents the energy needed to complete a task, static energy is parasitic and only poses an overhead on the computation. Although leakage can be mitigated in standby mode using techniques such as power gating and body biasing, such mitigation is more difficult to do in active mode. Hence, leakage forms an unavoidable and fundamental limit on energy efficiency.

Initially, when V_{DD} is large relative to V_t, frequency scales proportionately to V_{DD}. As V_{DD} is further reduced and nears V_t, frequency scales exponentially with V_{DD} because the transistor is no longer fully activated. Instead, the transistor drive current comes from subthreshold leakage current, which scales exponentially with the gate-to-source voltage and, thus, exponentially with V_{DD}. As operating voltage is lowered, the static energy increases, because the time to complete a task scales inversely with clock frequency. Eventually, at very low voltages, static energy dominates over dynamic energy (see Figure 1), as simulated in Cadence Spectre with industrial transistor models. The canonical circuit topology was a chain of 31 fan-out of 4 inverters along with dummy devices for realistic input and output slew rates. We chose a logic activity factor of 15 percent to emulate a core where 15 percent of the logic gates switch on average per clock cycle.[11]

The operating voltage where total energy is minimized is called V_{opt}, and it occurs when the derivatives with respect to V_{DD} of the two energies are equal: $dE_{\mathrm{static}}/dV_{\mathrm{DD}} = dE_{\mathrm{dynamic}}/dV_{\mathrm{DD}}$.[10] Beyond this point, static energy increases more rapidly than dynamic energy decreases, and the total energy increases away from the energy minimum. For a 32-nm node, V_{opt}, when considering leakage overheads, is ~300 mV or ~100 mV above V_t.

To fully compensate for a frequency loss of X due to reduced voltage operation, a task with k instructions must be parallelized across X cores. If frequency were not compensated for, the total execution time

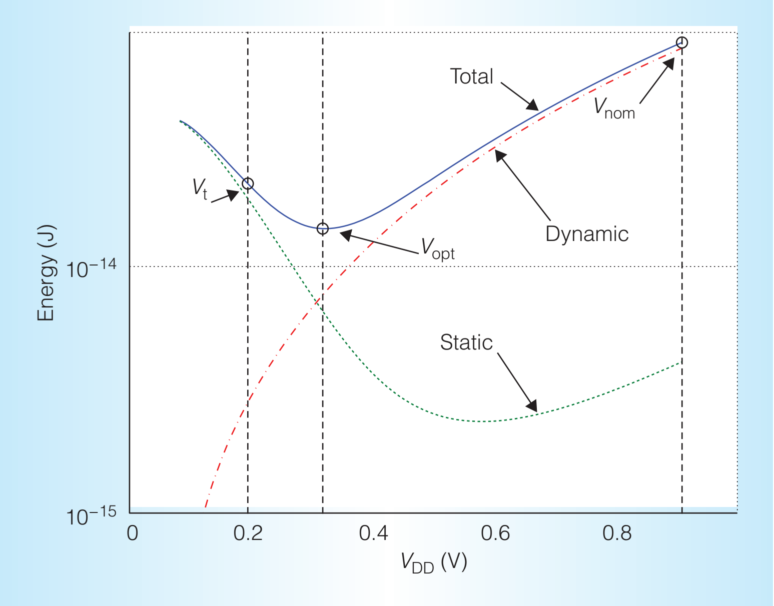

Figure 1. Total, static, and dynamic energy across V_{DD} for a 32-nm process. At very low voltages, static energy dominates over dynamic energy.

would increase proportionally to $X * k$. But, because the task is parallelized, each of the X cores runs k/X instructions, so the total execution time is $X * (k/X) = k$. Thus, no performance is lost from parallelizing. Although most scientific and high-performance applications have been parallelized to operate on CMPs, it is not practical to recover a factor of 100s or 1000s in frequency loss without enormous parallelization overheads. Therefore, V_{opt}, when considering optimization overheads, will be at a higher voltage level, which we will discuss in the next subsection.

Amdahl overheads

As discussed previously, scaling voltage is essential in the CMPs to achieve maximum computational performance for a fixed thermal budget, because the number of simultaneous tasks that can fit in a thermal design power (TDP) is directly proportionate to the task's energy efficiency. When scaling supply voltage for a latency-sensitive task, executing the task in parallel across more cores can compensate for slower clock frequency.

For real applications, the process of subdividing a task includes nonidealities, such as serial portions of code, and thus incurs parallelization overhead. Compensating a task of k instructions for a frequency loss of X requires X cores (each running k/X instructions) plus m additional instructions of parallelization overhead. These extra m instructions consume additional energy, penalizing lower-voltage operation, and thus increase V_{opt}. Hence, parallelization overheads compound the impact of leakage overheads, which limits a latency-sensitive task's voltage scalability. By Amdahl's law,[12] speedups are bounded asymptotically as parallelization increases because of serial portions of code. These overheads will be referred to as "Amdahl overheads" and include only the impacts of algorithmic parallelization.

We used the gem5[13] system simulator to evaluate the impact of Amdahl overheads on a network-on-a-chip (NoC) system. Additionally, we used the Splash-2 benchmark suite, which is a set of highly parallelized scientific algorithms. Each core is an Alpha architecture with one instruction per cycle running at 1 GHz. To separate Amdahl overheads from additional architectural nonidealities, we simulated the system with infinite interconnect bandwidth and an ideal memory with 1-cycle latency.

We fitted the Splash-2 benchmark speedups to Amdahl's law and applied the law to the voltage-scaling calculations to obtain V_{opt} when considering nonideal parallelization. The benchmarks were parallelized to fully compensate for frequency loss from lower-voltage operation. Figure 2a shows V_{opt} increasing in 32 nm because of Amdahl overheads. Some Splash-2 benchmarks, such as Barnes, have nearly ideal speedup, indicating very little parallelization penalty from Amdahl overheads. Other benchmarks, such as LUN, reach a speedup of only 10× with 64 cores, indicating a high percentage of serial code. These benchmarks represent a range of parallelized scientific workloads applicable to CMPs. When Amdahl overheads are added, the V_{opt} operating range for most overheads is ~25 mV to ~150 mV above the leakage-overheads-only case. Although the serial coefficient is highly application-dependent, the range of V_{opt} for the benchmarks is small, varying by only ~150 mV. If the serial coefficient were 100 percent (that is, if none of the code were parallelizable), then nominal voltage would be optimal.

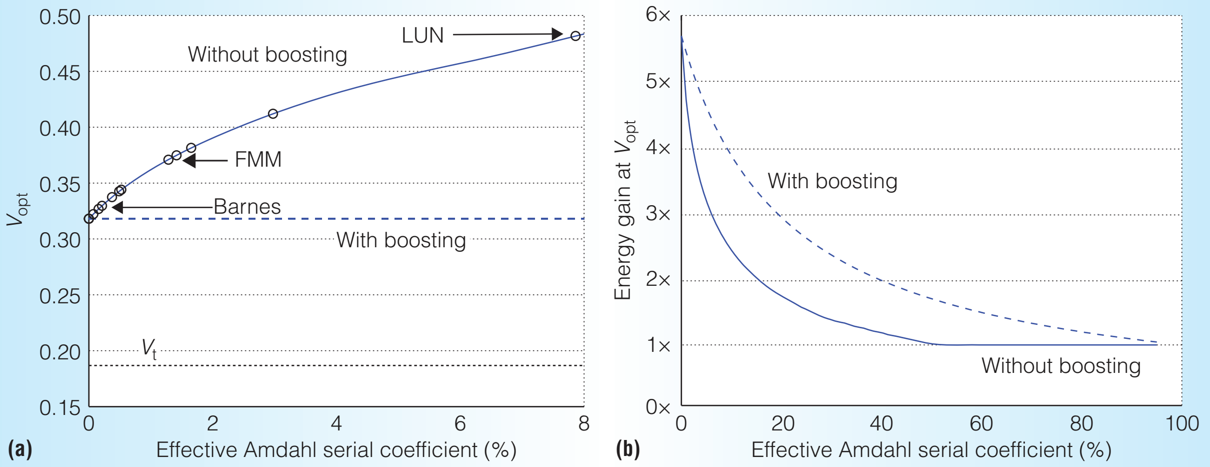

Figure 2. Examining the Amdahl overheads. V_{opt} versus Amdahl coefficient for all Splash-2 benchmarks (three labeled) in 32 nm (a). Energy gain by operating at V_{opt} versus Amdahl coefficient (b).

If the serial code portion could be efficiently detected in a parallelized algorithm, then the system could operate between two simultaneous voltage modes, depending on whether an algorithm's serial portion or a parallel portion were running. In a system equipped with voltage boosting,[14-16] where a single core's operating voltage can be rapidly increased for high-performance single-threaded operation, then the serial portion can be quickly overcome, regaining parallelism speedup and extending the number of cores for which an algorithm can be further parallelized. Recent work has shown that cores can be boosted within approximately 9 ns, or roughly one cycle, when operating at low-voltage clock frequencies.[16] We modeled this case, by assuming that the serial code portion can be perfectly separated from the parallel portion and that it is run on a dedicated boosted core operating at full voltage. Because frequency loss from voltage scaling of a serial portion cannot, by definition, be compensated by parallelizing, the serial code runs most efficiently at full voltage. Similarly, the parallel portion now runs most efficiently at the leakage-only V_{opt}, because all serial portions have been removed. Thus, V_{opt}'s dependence on the Amdahl serial coefficient is flat in Figure 2. Of course, serial code cannot be perfectly separated from parallel code, and in a real system nonidealities would increase V_{opt} somewhere between the two "without boosting" and "with boosting" extremes.

In 32 nm, energy gain by operating at V_{opt} decreases from nearly 6× to 1× as the Amdahl serial coefficient increases from 0 percent to 100 percent (see Figure 2b). Without boosting, energy gains decrease dramatically as the Amdahl coefficient is increased and, with a 15 percent coefficient, energy gain is only 2×, or one-third of the ideal gain. Additional energy can be recovered by introducing boosting, where at a 15 percent Amdahl coefficient, the energy gain is 3.3×. At an Amdahl coefficient of 50 percent, no energy can be gained by parallelizing and operating at V_{opt} without boosting, while boosting can recover 63 percent more energy.

Architectural overheads

Architectural features, such as coherency, intercore communications, and cache pollution, further add overhead to a CMP system as voltage is reduced and a task is parallelized. Furthermore, application memory access patterns can affect overhead. For example, a subtask competes for L2 cache resources and can evict another subtask's data. Coherence overhead is added when multiple subtasks share a single block of data. Communication overhead is increased when there is heavy communication between distant cores on an NoC, because data must transverse multiple hops.

Table 1. Simulation parameters for measuring architectural overheads.

Feature	Description
Cores	1 to 64 one-IPC Alpha cores @ 1 GHz
L1 caches	32 Kbytes, 1-cycle latency, four-way associative, 64-byte line size
L2 caches	Shared 1 Mbyte divided evenly between cores, 10-cycle latency, eight-way associative, 64-byte line size
Interconnect	128 bits, two-stage routers @ 2 GHz, 50-cycle access to main memory

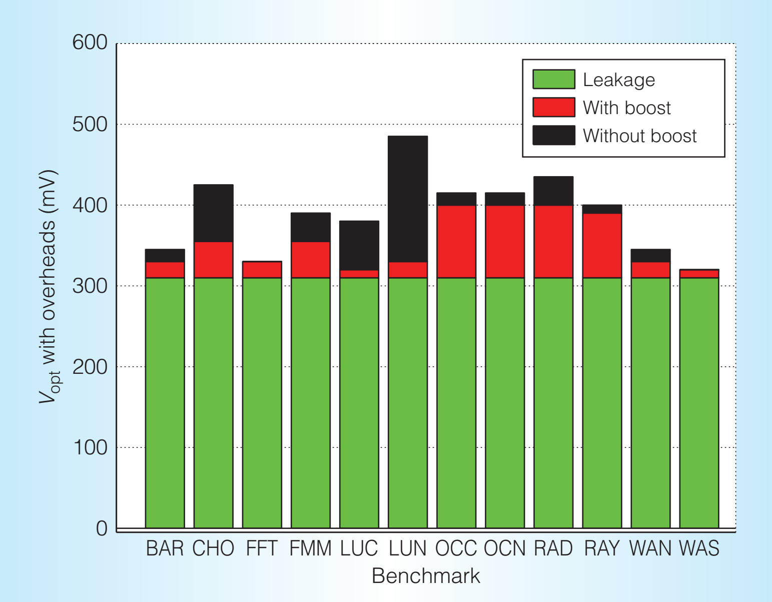

Figure 3. Architectural and Amdahl overheads increase V_{opt} by no more than 100 mV with boosting and 200 mV without boosting. Leakage, Amdahl overheads, and architectural overheads increase the minimum energy consumption as well.

To quantify architectural overheads, the Splash-2 benchmarks were simulated with gem5 as in the previous section, but the configuration was changed to add nonideal memories, caches, and interconnects. The NoC simulations were run using a tiled mesh topology in which each tile contains a core, private L1 caches, and a slice of a shared L2 cache. A MOESI directory protocol is used to maintain coherence. Table 1 lists the detailed simulation parameters.

Architectural overheads from memory and interconnect nonidealities reduce the obtainable speedup when parallelizing. These nonidealities were added in the V_{opt} calculation when parallelizing, and the benchmarks were again parallelized to fully compensate for frequency loss. Like leakage and Amdahl overheads, architectural overheads further increase the minimum energy consumption and V_{opt}, as shown in Figure 3.

Unlike with Amdahl overheads, we did not assume architectural overheads could be perfectly separated into serial and parallel phases and overcome through boosting. As a comparison, Figure 3 includes the additional impact of Amdahl overheads on V_{opt} for architectures without boosting. Certain benchmarks are highly parallelizable before caches and coherency are introduced, while others have negligible architectural overheads. For example, the OCN benchmark has almost no Amdahl overheads but significant architectural overheads. In contrast, the LUN benchmark has little architectural but significant Amdahl overheads. Across the benchmarks shown, V_{opt} increases by no more than 100 mV with boosting and 200 mV without boosting. Thus, architectural overheads are another key limiter to voltage scaling, increasing V_{opt} and the minimum obtainable energy consumption when a task is parallelized to compensate for frequency loss.

Impact of technology and circuit features on NTC

The previous section discussed the three key limiters of energy-efficient scaling. However, V_{opt} is also impacted by additional technology and circuit factors, including technology node. We discussed additional technology factors, such as transistor V_t and process variation, in a previous article.[7]

Technology

In the previous section, V_{opt} was analyzed at single 32-nm technology node. To identify whether a voltage-scaling and parallelization guideline is consistent across many technologies, we calculated V_{opt} for Splash-2 across six industrial technologies when accounting for all three voltage-scaling overheads, as shown in Figure 4. Circuit simulations of energy and performance were done in Cadence Spectre using industrial foundry technology kits from 32 nm to 180 nm.

The process node affects V_{opt} primarily because technologies have become more

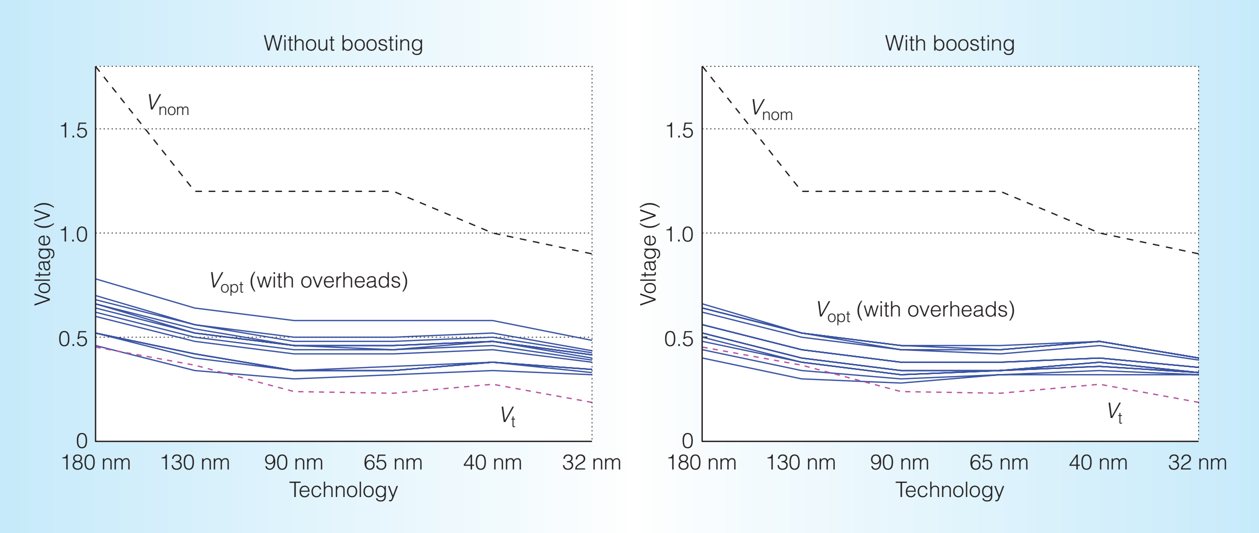

Figure 4. V_{opt} across technologies when including all three overheads without boosting (left) and with boosting (right). V_{opt} tracks ~200 mV to ~400 mV without boosting and lowers to ~100 mV to ~200 mV with boosting.

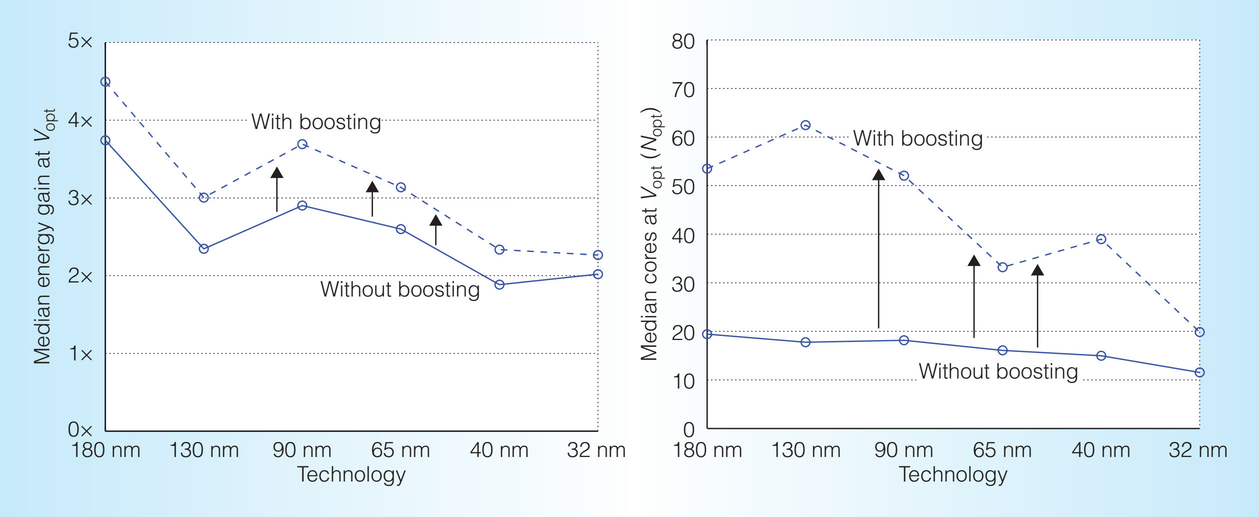

Figure 5. Median energy gains and optimal number of cores, N_{opt}, when operating at V_{opt} as compared to nominal voltage for Splash-2 benchmarks. Achievable energy gains and N_{opt} decrease with smaller feature sizes.

leaky with each generation due to reduced threshold voltage. Higher leakage increases V_{opt}; however, the lower threshold voltage will also improve the frequency degradation with voltage scaling, which will reduce V_{opt}. A key finding of this work is that, for most benchmarks across the six technology nodes, V_{opt} consistently tracks ~200 mV to ~400 mV above the threshold voltage without boosting and ~100 mV to ~200 mV with boosting. We define this region above the threshold voltage as the near-threshold computing (NTC) region. Three benchmarks—Barnes, FFT, and Water Spatial—were close to ideally parallelizable and are not contained in the NTC region. However, most general-purpose, high-performance CMP applications will have some degree of parallelization overhead and thus lie in the NTC region.

Figure 5 shows the median energy gains at V_{opt} operation, with and without boosting, and the optimal number of cores to parallelize across to compensate for clock frequency loss,

N_{opt}, for Splash-2 across technology nodes. Energy gains have diminished by ~1.8× from 180 nm to 32 nm as leakage has increased and the dynamic range available for voltage scaling has narrowed from 180 nm to 32 nm. This difference is less dramatic with less-scalable benchmarks, because the parallelism overheads are higher and thus the amount of voltage scaling in older technologies is limited.

In newer technology, the energy gains from operating at V_{opt} without boosting instead of at nominal voltage are ~4×, and N_{opt} has a median of ~12, and no more than 25, cores for Splash-2 in 32 nm without boosting. With boosting, the energy gain in 32 nm is 4.5×, while the median number of cores is ~20 and no more than 25 cores. However, gains by boosting are expected to improve for benchmarks that are less parallelizable, with an Amdahl coefficient above 10 percent, but with negligible architectural overheads. This increased energy efficiency directly increases CMP performance when limited by a thermal budget. Thus, to maximize thermally limited CMP performance, tasks should operate in the NTC region and parallelize on no more than 25 cores with and without boosting. The energy gains and optimal amount of parallelism has decreased with each generation.

We have detailed the limits of voltage scaling for latency-sensitive applications, when slower clock frequency is compensated by parallelization across multiple cores. As CMPs become limited by thermal-cooling constraints, near-threshold operation is needed to maximize the computations for a fixed thermal design power. However, many obstacles remain before near-threshold designs can be fully realized in commercial systems. Process variation and supply-noise sensitivity can be very high in the near-threshold region. Increased clock skew and hold-time uncertainty further inflate timing margins, limiting clock frequency and achievable energy gains. These effects, however, can be mitigated through soft clocking and in-situ error detection techniques.[17] Additionally, a fundamental challenge of boosting a core between near-threshold and super-threshold supply voltages is circuit scalability. For example, fewer repeaters are required for an on-chip interconnect in near-threshold than in super-threshold design, because wire delay becomes relatively faster compared to circuit delay as V_{DD} is reduced. Thus, design optimizations to improve performance in near-threshold may negatively affect super-threshold performance. Developing techniques to minimize super-threshold impact is critical for realizing a high-performance near-threshold system. MICRO

Acknowledgments

The authors gratefully acknowledge the National Science Foundation, ARM Holdings, and DARPA (agreement #HR0011-13-2-0006) for their support of this project.

References

1. S. Sawant et al., ''A 32 nm Westmere-EX Xeon Enterprise Processor,'' *Proc. IEEE Int'l Solid-State Circuits Conf.* (ISSCC 11), IEEE CS, 2011, pp. 74-75.
2. H. Esmaeilzadeh et al., ''Dark Silicon and the End of Multicore Scaling,'' *Proc. 38th Ann. Int'l Symp. Computer Architecture* (ISCA 11), ACM, 2011, pp. 365-376.
3. G. Moore, ''No Exponential Is Forever: But Forever Can Be Delayed,'' *Proc. IEEE Int'l Solid-State Circuits Conf.*, IEEE CS, 2003, pp. 20-23.
4. Bo Zhai et al., ''Energy Efficient Near-Threshold Chip Multi-Processing,'' *Proc. ACM/IEEE Int'l Symp. Low-Power Electronics and Design* (ISLPED), 2007, pp. 32-37.
5. M.B. Taylor, ''Is Dark Silicon Useful?: Harnessing the Four Horsemen of the Coming Dark Silicon Apocalypse,'' *Proc. 49th Ann. Design Automation Conf.* (DAC 12), ACM, 2012, pp. 1131-1136.
6. A. Chandrakasan, S. Sheng, and R. Brodersen, ''Low-Power CMOS Digital Design,'' *IEEE J. Solid-State Circuits,* vol. 27, no. 4, 1992, pp. 473-484.
7. N. Pinckney et al., ''Assessing the Performance Limits of Parallelized Near-Threshold Computing,'' *Proc. 49th Ann. Design Automation Conf.* (DAC 12), ACM, 2012, pp. 1147-1152.
8. A. Wang and A. Chandrakasan, ''A 180 mV FFT Processor Using Subthreshold Circuit Techniques,'' *IEEE Int'l Solid-State Circuits Conf.*, IEEE CS, 2004, pp. 292-529.

9. B. Zhai et al., ''Theoretical and Practical Limits of Dynamic Voltage Scaling,'' *Proc. 41st Ann. Design Automation Conf.* (DAC 04), ACM, 2004. pp. 868-873.
10. L. Nazhandali et al., ''Energy Optimization of Subthreshold-Voltage Sensor Processors,'' *Proc. 32nd Ann. Int'l Symp. Computer Architecture* (ISCA 05), ACM, 2005, pp. 197-207.
11. J.M. Rabaey, A.P. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A Design Perspective,* Prentice Hall, 2003.
12. G.M. Amdahl, ''Validity of the Single Processor Approach to Achieving Large-Scale Computing Capabilities,'' *Proc. AFIPS Conf.,* ACM, 1967, pp. 483-485.
13. N. Binkert et al., ''The gem5 Simulator,'' *ACM SIGARCH Computer Architecture News,* vol. 39, no. 2, 2011, pp. 1-7.
14. ''Intel Turbo Boost Technology in Intel Core Microarchitecture (Nehalem) Based Processors,'' white paper, Intel Corp., Nov. 2008.
15. A. Raghavan et al., ''Computational Sprinting,'' *Proc. IEEE 18th Int'l Symp. High Performance Computer Architecture* (HPCA 12), IEEE CS, 2012, doi:10.1109/HPCA.2012.6169031.
16. R.G. Dreslinski et al., ''Reevaluating Fast Dual-Voltage Power Rail Switching Circuitry,'' presented at 10th Ann. Workshop Duplicating, Deconstructing, and Debunking, 2012.
17. R.G. Dreslinski et al., ''Near-Threshold Computing: Reclaiming Moore's Law Through Energy Efficient Integrated Circuits,'' *Proc. IEEE,* vol. 98, no. 2, pp. 253-266.

Nathaniel Pinckney is a PhD student in electrical engineering at the University of Michigan. His research interests include low-power VLSI design and cryptographic accelerators. Pinckney has an MS in electrical engineering from the University of Michigan.

Ronald G. Dreslinski is a research scientist at the University of Michigan. His research focuses on architectures that enable emerging low-power circuit techniques. Dreslinski has a PhD in computer science and engineering from the University of Michigan.

Korey Sewell is a senior engineer at Qualcomm. His research interests include on-chip interconnect and memory controller design, high-performance microprocessors, many-core systems, and simulation modeling. Sewell has a PhD in computer science and engineering from the University of Michigan, where he performed the work for this article.

David Fick is a research associate at the Michigan Integrated Circuits Laboratory, within the Department of Electrical Engineering and Computer Science at the University of Michigan. His research interests include fault tolerance, adaptive circuits and systems, and 3D integrated circuits. Fick has a PhD in computer science and engineering from the University of Michigan.

Trevor Mudge is the Bredt Professor of Engineering at the University of Michigan. His research interests include computer architecture, programming languages, VLSI design, and computer vision. Mudge has a PhD in computer science from the University of Illinois. He is a fellow of IEEE and a member of the ACM, the IET, and the British Computer Society.

Dennis Sylvester is a professor in the Department of Electrical Engineering and Computer Science at the University of Michigan, and the director of the Michigan Integrated Circuits Laboratory. His research interests include the design of millimeter-scale computing systems and energy-efficient near-threshold computing for a range of applications. Sylvester has a PhD in electrical engineering from the University of California, Berkeley. He is a fellow of IEEE.

David Blaauw is a professor in the Department of Electrical Engineering and Computer Science at the University of Michigan. His research focuses on VLSI design, particularly on ultra low-power and high-performance design. Blaauw has a PhD in computer science from the University of Illinois at Urbana-Champaign. He is a fellow of IEEE.

Direct questions and comments about this article to Nathaniel Pinckney, 1301 Beal Ave., Ann Arbor, MI 48109; npfet@umich.edu.

cn *Selected CS articles and columns are also available for free at http://ComputingNow.computer.org.*